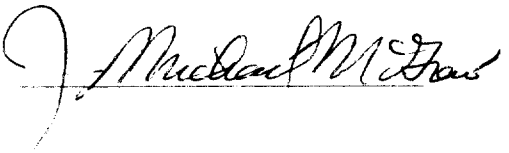
From IBM To VAX

An Honors Thesis (HONRS 499)

by

Julie A. Collis

Thesis Advisor
J. Michael McGrew



Ball State University

Muncie, Indiana

December, 1991

Expected date of graduation: May, 1991

Purpose of Thesis

The Department of Natural Resources wanted a program called Hothouse to be made available on the VAX system, in addition to the current version they have running on IBM PC's. The process of making a program identical to the original, but on a different system, is called porting. In other words, Hothouse is to be ported from IBM to VAX. Originally, the porting was attempted through the language of BASIC, but eventually the program was ported thrcugh pascal. Graphics were completely redrawn using ReGIS.

ACKNOWLEDGEMENTS

Thanks to the students, alumni, faculty and Honors College staff who assisted in the preparation of the guide I used when completing my honors thesis. Especially useful advice for programming help came from Bryan Strawser, Alan Hill, Matt Stum, Paul Neubauer, and my advisor, J. Michael McGrew.

Thursday, August 22

Today was my first day at work. I met Paul, my immediate supervisor, and the other staff members at University Computing Services. Paul informed me about the project I would be working on, which is called Hothouse. Hothouse is a copyrighted program that is written in some form of basic on an IBM system. The Natural Resources department at Ball State University uses this program. This department would like the program placed onto VAX so that it is more easily accessible. Moving a program from one system to another is called porting, which is what I am supposed to do with Hothouse.

Because Hothouse is written in some form of basic on the IBM, Paul would like the program to be ported in BASIC because the two languages are fairly similar. I have never worked with VAX BASIC or porting, so this should be quite a task. Although, I signed out three manuals on VAX BASIC to my desk, which should serve as good references.

Friday, August 23

I looked at the program a little more in depth today. I noticed quite a few "CLS" commands in the program. That is the clear screen command on a PC, and it does not work on vax. I changed all the CLS commands in the program to the escape sequence on the vax which clears the screen. In addition, I defined ESC as chr(27) for future use.

I also found a locate command in the program. This command finds a position on the screen. Immediately following each locate command was a print command, which allowed sentences to appear at that position on the screen. I could not find Paul, so I asked a few people in the office how to get the sentence to the correct position. I was told to press the space bar for the amount of spaces needed and to get rid of the locate command. This was tedious work which took me until the end of the day to finish.

When I was finished, I spent some time looking at the rest of the code for the program. I never got a chance to work on it more.

Monday, August 26

Early this morning I met with my advisor, Dr. McGrew. I asked him if I could have corrected the locate problem easier. He told me and helped me start writing an ANSI sequence to perform the task.

Locate was a reserved word, so I used the command *s/locate/location/begin:end to change all the locate commands to location. I finished writing the ANSI escape sequence, by making it fit the VAX BASIC format. Everywhere location occurred, other than where it was initially defined had to be changed to Call Location. The reason location needed to be changed to call

location is because I made location a subroutine. Then, from each call location, I passed the values to the subroutine.

The next problem that occurred had to deal with the colon. In between each locate and print command, there was a colon which meant "do the second task after performing the first" on a PC. VAX would not accept the colon. Therefore; I had to go through the program, delete all the colons after locate commands and then press return so that locate and the next command were on two different lines. If a print statement was on the second line, I indented it for appearance and so that it was easier to read.

Tuesday, August 27

For the first hour of work today, I kept trying to compile the program. When the compiler gave me errors, I corrected them. The last error for the hour said "BLOAD.PIC." On the PC, this is a command to load a picture in binary form onto the screen.

I asked Paul about what we were doing for graphics. He told me to check and see what type of graphics we could use. I checked around with board operators, Matt Stum, Bryan Strawser, Gary Gressel, and some other programmers. After questioning a number of people, I found that REGIS graphics would best suit the needs of this project. I had never worked with REGIS, so Paul took me to Jolanta to get a book on REGIS. I read through the manual (about ten pages) and figured I could probably figure it out as I went along.

I found a piece of graph paper, looked at the IBM screen, then drew a sunlight picture freehand. I tried to use REGIS and it was fairly simple. Although, the graphics were not as precise as I wanted them to be. I found out from another intern that you could get more precise if you wrote out the commands yourself describing each pixel rather than using a pre-set-up screen. For the last couple of hours at work, I learned how to use it, began drawing the first picture.

Wednesday, August 28

Paul managed to print out copies of all the pictures so that I could place a piece of graph paper over his picture for more accuracy. I no longer had to look at the screen and guess where on the graph paper that a line, circle, or object lied. Scaling was made a lot easier too.

I noticed that three of the graphs were very similar, so I decided to work on one of these first so that I could copy that picture twice, then modify the two copies. The first picture (graph) was fairly easy to draw. For me, getting the text onto the picture was the hardest part. I had never put text on a picture before, so I did not know those commands. Also, the text did not match up exactly with the positions on the graph paper.

When I finished the picture, I copied it twice. I then changed the text on the two new graphs to what it needed to be, and I raised the bars (or lowered them) to where they were

supposed to be. The three graphs were finished just as I got off work.

Thursday, August 29

Not much was accomplished today. In the three hours that I worked, only one full picture was finished. The picture that I finished was a line graph over a one hundred year period. I attempted to draw a picture of a volcano several times, but every attempt was a failure. On my first attempt, I used the version of REGIS that was not as precise (GE commands) -- accuracy was not good enough for me to use this version. The second time, I used hand-positioning in REGIS. This too failed because I was unable to shade the lava, and smoke correctly. I do not know how I will get this picture drawn and functioning properly.

Friday, August 30

As soon as I arrived at work I typed BAS hothouse, to see how many errors were in the program I was working on. There were 853 errors. All that I did all day was work on lowering the amount of errors in the program. By the end of the day, there were only 208 errors. Below is list of some errors and how they were corrected:

1. line(x,4)-(x,4) : On IBM PC
   I typed print "------" to have the word underlined.
2. Put (x,4) two % : on IBM PC
   I commented these out -- they are subscripts which will be dealt with later.
3. LPRINT "Message" : on IBM PC
   I commented this out -- not sure if the instructor will want students to print graphics out or not.
4. Circle (x,4),2 : on IBM PC
   If it was a graphic, I deleted it because I redrew the graphics. If it was a degree, I commented that out for use later.
5. : : on the IBM PC
   An actual colon on the IBM. I split the statements on two separate lines and provided spacing as needed.
6. For/next i,j : on IBM PC
   Made it next i and next j, and put it on two separate lines. Then I split the for/next loops up as needed and provided spaces.

Monday, September 2

There was no work today -- Labor Day.

Tuesday, September 3

As I tried to correct more and more errors, I found that variable names were not descriptive enough. Therefore; I changed

the variable names to something more appropriate after figuring out what the variables were. In case I had misunderstood a variable's purpose, I also wrote the old names, and what I had renamed them to in a file called arrays.txt;. Then, I spaced the program out as needed. When it was spaced in a clearer manner, I noticed some unnecessary lines, and deleted them. Two errors were corrected:

1) Everywhere an "if" statement appeared, at the end of the clause, I needed to add an "end if ".
2) Gosubs could only call items above where they appeared, so I needed to move code called below to an area above.

Wednesday, September 4

I finally corrected all the errors in the program that the computer detected, with the exception of one out-of-range error in "glossary". I still can not locate that error. I tried renaming glossary, but it failed. After trying to fix glossary for about an hour, the head honcho, Bizhan Nasseh came to see my "two week progress." He saw my work and stated that I had done good work and accomplished a lot in two weeks. When Bizhan left, I drew the outline and did the text for four more graphics that needed to be drawn. I spoke with Paul, who informed me that we are going to use ASCII graphics with all terminals that do not accept REGIS. Terminals whose numbers are two hundred or above generally accept REGIS. Tomorrow, I will redraw the REGIS graphics using standard characters for the non-REGIS capable terminals.

Thursday, September 5

I did not work today because I was sick.

Friday, September 6

I can hardly believe the enormous amount of work I completed today. All four graphs that I drew using REGIS were redrawn using ASCII characters today. They are not nearly as nice as the REGIS ones, but it is the best that can be done for computers that are not REGIS capable. I also created characters for characters that do not exist on a keyboard. I replaced the 1,^,' and ~ with a 2,4,(both are used for subscripts), delta sign, and degrees sign. Then I went into the hothouse, deleted the "put" commands and replaced it with a subtwo$, subfour$, delta$, or degree$, depending upon what was needed. Then I tried to correct the glossary error I had been working on. I compressed lines in that area, where possible, until it was time to leave work.

Monday, September 9

I started out by compressing the glossary section, and

getting rid of a goto called PressKey. I hoped this would get rid of the glossary error, but it did not. Then, I moved all the data together, as it was scattered throughout the program. One section that previously was !LPRINT (in other words commented out) was deleted. The program was still in much disarray, so I tried to move things around, and delete unneeded sections. I went back through the program and found where the data was being read in from. Some of the data was not being used after being read in, so I deleted that data and the part where it was read in. The Glossary error was still out of range. I deleted a few more unneeded lines, and commented some out to find the glossary error. This time, the compiler detected the same type of error that was in glossary, in the quit procedure.

Tuesday, September 10

I still wanted to get the "out of range error" corrected. I had a feeling that the distance between the first line of the program and the quit procedure was to far apart; therefore, I decided to make all the gosubs into subroutines. I read the manual for the syntax on subroutines. Everywhere I saw a gosub, I changed it to call followed by the procedure name. Then, I put all subroutines at the bottom of the program. If a subroutine was called HowItWorks, then it became sub HowItWorks. The call to HowItWorks from another procedure would be call HowItWorks (parameters). Also, instead of a return statement, I needed an "end sub". I finished doing the subroutines, but did not have a chance to see if the program ran because it was time to leave.

Wednesday, September 11

After I arrived at work and logged on, I discovered that the program still had the destination out of range error. My only choice was to modularize the program to remove the error. Unfortunately, I was familiar with the editor EDT, which does not allow you to get a split screen. Therefore; it was very hard to remove sections and add them to a new file. Using EDT, you have to copy the entire program over. Then delete all but the sections you want. After that, you go back to the main program, and delete the section you did not erase. Today, because of this problem, I learned a few basic commands on the EVE editor. Not only did I learn how to get into EVE, but I learned that the DO button controls most commands. For example, DO split <return>, divides the screen into two sections. Then DO OPEN FILENAME, will open another file. You can then write to that file with DO WRITE. Control-Z will exit EVE and then you can write to the buffer. After learning these basic commands, I began modularizing the program.

Thursday, September 12

I did not work too long today. Although, I did complete a task. I completely modularized hothouse. Tomorrow I plan to give each module a more appropriate name than the module number, which is currently the name. For example, MOD2.BAS is the second module I created.

Friday, September 13

Wow! I actually made it through work without too much trouble, even though it is Friday the 13th. I did not rename the modules as I had planned to. Instead, I tried to find out how to sixelize my pictures so that they would run faster. I also tried to create a fill command to shade in parts of pictures. I wanted a fill command so that pictures would appear on the screen faster. It turned out to that the fill works on version 2.2 of VT240's, but not on a 1.3 version of a VT240. It looks like I am going to take three sets of pictures now rather than two. I already used the extensions, REG, and ASC, so I think PIC will be the extension for those computers that use fill. I also had to look up the command to determine what version each computer is, so that I can then have the program determine which extension to use. The reference was in the programmers' manual on page one hundred and two.

Monday, September 16

Well, a new problem arose today. Gotos can not be used for calling different modules. I went up to the forth floor and spoke with Delores Zage to try to find how I could get this to work. The answer is to remove goto statements, replace them with select/case statements or if/then statements, depending upon what suits the needs of the programs. Before being able to change these problems, I needed to give each module a more appropriate name so that I could find out which modules were being called from other modules. For example, MOD2.BAS became SUNLIGHT.BAS. Then, I went through each program and wrote down which subprograms the goto called.

Tuesday, September 17

I finished writing out my goto sheet, and then began combining modules if they were only called by one procedure. While doing this, I needed to space each added subroutine so that it was readable. I condensed the eighty files down to around fifty or so. I did not finish this task. Also, I had a slight problem -- the person that wrote the program tried to take the quick way out. Instead of using default clauses, the programmer made the program fall through to the next label. Therefore, I needed to find all the fall through spots. I did not have time to create the added default clauses yet.

Wednesday, September 18

I finished combining the program. Once combined, I tried to compile it. Unfortunately, the program did not compile. It gave me 124 block errors. The block errors said that I could not define a block within a block, meaning that in part of the procedure, it calls another part of that same procedure. It was hard to see each procedure on one screen, so I went downstairs and made printouts of what I had done, and the original program. I marked each spot where a block was inside another one. They were all ir for/next loops.

Thursday, September 19

Today, I attempted to fix the for/next loop errors. To correct some, I merely had to make a procedure into two procedures. Although, I had to watch how much space I was using so that I did not go back to the original problem of not enough space, which would give me an "out-of-range error." I got down to about sixty errors and then quit doing this for a while. Instead, I discovered that where there were two clauses and if the condition did not fit the clause, it dropped through rather than going onto a default statement. I went back to my hard copy and marked where everything was going to.

Friday, September 20

I went back in the program and put in a call to the procedures where it was dropping through on if statements. There were more than one hundred of these drops. When I compiled the program once again, I produced more block errors. I tried to correct a few of them by splitting procedures into two parts, but this time very few errors could be corrected that way. Frustrated. I decided work was done for the day.

Monday, September 23

When I arrived at work, I went to speak with Paul about the errors I was getting. After a long discussion, Paul and I agreed that it was more trouble to keep working with the BASIC version than to try to write the whole program over. Once again, I watched hothouse run on the PC. After playing with the IBM version for some time, I began typing the sentences that needed to appear on the screen, using Writeln('Statement');.

Tuesday, September 24

I finished typing in all the writeln statements. Then I typed in the command for clear screen (it was the same as the

basic command) everywhere that I knew it was used in the program. So far, the program compiled and ran. I wanted to make sure the program could run the REGIS graphics then. I wrote a procedure called DrawPicture that would read in lines from the picture files and then draw the picture on the screen. My procedure also worked, although the pictures were more of a slide show because they were not in correct spots. All pictures were at the end of the program. I realized that some of the pictures would reverse the X and Y axes or they would fowl up the next picture. Therefore; I found a reset terminal command and put it before each picture. Then all the pictures appeared correctly.

Wednesday, September 25

I thought that the next step in the program should be to put the words and pictures on the screen, in the order that they appeared on the IBM BASIC version of HOTHOUSE. There was one problem -- my boss was in the hospital and he was the only one with a copy of Hothouse. I put the words and pictures in the best order that I could, using the original program printout as a reference. I also added two CASE statements to the program. These two case statements are used in spots where the user is given an option to do one of several things. The case statement gets the choice and then runs that option. Each option became a procedure.

Thursday, September 26

When I started work today, the program was merely a bunch of words and pictures running across the screen. I wanted the program to be interactive, as the original version was. I put in a bunch of statements that said "Press Return to go on" followed by a Readln(Choice); statement. Any key could then be entered. As the program ran, I could see the layout of how some lines would appear on the screen. If I knew that certain things made up one page of a screen, then I centered them, as they were in Hothouse. If Paul was here, I could have verified these screens, but he was still in the hospital.

Friday, September 27

Did not work today -- went home to Michigan.

Monday, September 30

Did not work today -- went to court.

Tuesday, October 1

Did nct work today -- took care of some court matters.

Wednesday, October 2

Paul was back today. I got a copy of Hothouse in case anything like this would happen again. I kept running the program and comparing it to what I had done. Several items were out of order, but I corrected them. I found out where I needed more case statements too. Moving statements into the correct order used all of my work time up.

Thursday, October 3

At the bottom of each screen in Hothouse, there are arrows that say "<-- or -->". I needed to find out how to get these arrows to appear on the screen. I found out the best way was to type Writeln ("<-- or -->"). Then, I realized that I had to make sure that the user would type in a left or right arrow. I found out there is a program called GetKey that performs this task. I got a copy of GetKey and a call to it from my program, where needed. Basically, I used Repeat/Until statements so that my writeln statement keeps appearing until the user types a <-- or -->.

Friday, October 4

Today I added a few more arrow statements to the program. I also tried to find out how I could sixelize my graphics. Alan and I worked on sixelizing for our full time at work. We can get it to work for some pictures, but not others. We also could not figure out why only some pictures sixelized. We are going to keep trying to find a way to do this. After all, VAX is a time-shared system, so we want Hothouse to run as fast as possible.

Monday, October 7

The morning started off well. I was given a PC which I placed next to my VT240. Now I was able to see Hothouse run on both a VT240 and on the IBM at the same time. I began trying to make my version (on the VT240) look as much alike the version on the IBM as possible. First, I needed to draw the cover picture for the program. Of course, I added one line that said "Ported to vax by Julie Collis." I knew this line would be taken out later, but for the time, it humored me. Surprisingly enough, I found out that I still had a few things out of order. Everything now runs totally correctly through where a screen comes up and says "Main Menu." That means there are four screens before main menu that all run correctly. I discovered that it takes quite some time to make one version look exactly like the next. For now, I will work on getting a version running rather than on

speed.

Tuesday, October 8

When I arrived at work, I ran the PC version of Hothouse. Then I ran my version of Hothouse. I noticed that pretty much all that I had left to do was the six large sections under the Main Menu. Today, I first wrote the procedure for quitting, which I found to be very easy. I put a repeat/until loop around the main menu and then in the quit procedure, I set Y equal to a value so that it could exit if the user chose to do so. Also, I included a clause in case the person did not want to exit and accidentally pushed six. Section five was the glossary. I finished making this section work too. Both of these procedures are very robust. There are only four more sections left to be completed after the main menu screen.

Wednesday, October 9

Section one of the program was finished today, although, it is not nearly as robust as sections five and six. I added the arrows so that you can go to a forward screen of a backward screen. The volcano picture is not working quite correctly with the print on the screen. The sunlight picture is not totally complete either. To get the arrows to work, I nested a bunch of repeat/until statements within each other. I stated working on section three also. It needs a lot more work. Surprisingly, I still had a few mixed up items in that procedure, as far as order is concerned. I think it is all in order now.

Thursday, October 10

I finished section two in Hothouse, which consists of how the program works. It runs totally correctly. Although, if the Natural Resources department wants something changed in there, I will have to go back and do it. Alan and I worked on sixelizing the drawings for Hothouse. Alan was mailed a sixelizer from someone who saw his posting on Newsgroups. Some of the drawings run faster when they are sixelized, others are slower. It basically depends on the amount of text. If there is a large amount of text used, then it will run faster using REGIS rather than the sixelized version. If it is a highly detailed drawing, I found I am better off using the sixelized version. Also, different machines run the sixelized drawings at different speeds. That is something I am going to have to check into. I am also going to need to make a call to the computer to find out what type of terminal it is so that I know which version of the graphics to run. Section four is in order, but not running entirely correctly yet. The case statements runs, but then you are stuck in the section until a control-c is pressed. There will be one version of Hothouse working that the Natural

Resources department can use before it is speed efficient.

Friday, October 11

I wanted to see if there were any postings about Hothouse in news or if I could find anything in news that would work better for Hothouse. All day, I read the news, and I made a few postings. I had never really used news before, so now I am familiar with it. I spaced out a few lines in Hothouse so that they appeared nicer on the screen, but other than that, there was nothing more accomplished on the actual program today.

Monday, October 14

In part four, there is a section where the user can input an area code or a zip code for anyone in the United States. Once the number is entered, they can then see the statistics for that area. Today, I entered in the data for this section. I used a case statement for all of the numbers you could enter, then I assigned that number to a region. I did this because several regions have the same data screen. Once I had created the large case statement, I stored the data for the data screens.

Tuesday, October 15

Yesterday's work was repeated today. The only difference was that I was working on data for Canada today, using postal codes rather than zip codes. The entire section for Canada is set up in the same manner as the United States' is.

Wednesday, October 16

The data in section three did not appear in a neat enough manner to me when I ran the program today. Therefore; I decided to use REGIS to make the data appear in a neater fashion. I created the outline for the first screen. Then I plugged in the first screen's data, which happened to be for carbon dioxide. After I plugged in the data and it looked nice, I copied the picture file five times. Then I replaced the data for carbon dioxide with data for the other items in the program: nitrous oxide, solar irradiance, methane, and volcanic aerosols. These new screens are much more comforting than the old ones.

Thursday, October 17

After seeing how nice the other REGIS data screens were, I deleted the minimum, maximum, and contemporary values data. Then I recreated the files using REGIS. These screens are seen in section three.

Friday, October 18

Even though I had drawn out several data screens, I had never actually made the program so that it could use these screens. As a result, I went into Hothouse, found where the screens occurred, then added DrawPicture('RegisFileName');. A few other changes were made too. For example, I had to put arrows at the bottom of some pictures.

Monday, October 21

Today Paul downloaded three pictures for me. One was a map of the United States which would be used in several sections. The other two were world maps. The only difference between the latter two is that one was just an outline, and the other was shaded in. For my purposes, I thought the filled in picture would be better. I sixelized all three drawings though. I also added spots to the program so that it was accessing these picture files.

Tuesday, October 22

After running the program several times, I discovered that for the sixelized pictures, one line was always printed then deleted. To fix this problem, I had to go into the DrawPicture procedure, and change the readln statements. The problem was that in a loop, I was reading data, then writing it to the screen. I needed to do a readln before the loop, then a writeln before the readln in the loop. Before, the program was going past end-of-file. Now it is running correctly.

Wednesday, October 23

I ran the program again to find more problems. There were not arrows at the bottom of all the screens that needed them. Therefore; I added the "<-- or -->." Also, I positioned the arrows so that they appeared nicely at the bottom of the screen.

Thursday, October 24

Paul informed me that on section three, the first part where you can choose numbers was not working. I had never noticed this problem before. Today I put loops in that section so that if you hit R or r, the program would give you the screen again. If a number was hit, it would appear on the screen in place of the other number. If you hit return, a new number would appear.

Friday, October 25 through Friday, November 1

I did not work this week because I was sick with a respiratory infection and sinus infection.

Monday, November 4

The volcano picture that I had put off for a long time was finished today. I used Alan's terminal so that I could use the fill command. Once it was sixelized, I could see it on my terminal. I created smoke by defining @, #, and ~. To do this, you draw an eight by eight grid. Then, fill in dots where you want them in that eight by eight grid. When finished, divide the left to right eight into four and four. Then where a dot appears, write one, and where a blank space appears, write zero. Then figure out what the hexadecimal number is for that area. After getting all sixteen hexadecimal numbers, going from left to right, then down, write the sequence again. This is what I defined each symbol of @, #, and ~ by. Once finished, I positioned the cursor where I wanted the smoke, chose my colors, and printed out the character. I think this picture took the most time of them all. It looks fairly decent when you run the sixelized version.

Tuesday, November 5

I put off drawing the picture of Atlantic City for quite some time. Today, it was completed. Again, to make the programming a little easier, I used Alan's terminal. The Atlantic City picture consisted of what it currently appears as with land, marsh, and water. The picture to the right of it shows what it would look like with a rise in water.

Wednesday, November 6

Gulfport was another picture similar to Atlantic City. I did this picture in the same fashion that I did the previous one.

Thursday, November 7

Charleston was the same type picture as the two preceding ones. It was completed in the same manner as the other two. I also created blank ASCII files for these picture files.

Friday, November 8

VAX was not up during the period I was working today, so I did not complete any more work on Hothouse.

Monday, November 11

Previously, section three was running only on the GetKey routine. The arrows were not functioning. I nester repeat/until statements so that the arrows functioned correctly. If Special was equal to a right arrow, then it went to the right, otherwise, it checked where it was in the repeat/until loop.

Tuesday, November 12

I found information on how to make Hothouse run if the terminal was REGIS capable. I wrote the routine, checking for a one or a zero under the REGIS category of DVI. Then, if the terminal was REGIS capable, the program ran. If not, I wrote a note to the user that he could not use this program on the terminal he was on, that he would need to be on a VT200 or higher numbered terminal.

Wednesday, November 13

Section four was missing some words and subscripts, which I promptly added. It was also slightly out of order. Once in order, I went through each section and fixed any minor errors, like spelling, that I had found.

Thursday, November 14

At the beginning of Hothouse, the user is given a choice of using United States units of measurement, or metrics. However, this information had never been used. Today I put in the conversion factors and also made it so that if the user typed M for metrics, the numbers would all appear in centimeters, and other metric units. Otherwise, it would appear in inches and other United States units. MorU was the variable name. I passed this value to all parts of the program needing it.

Friday, November 15

I was concerned with getting small errors out of the program. One of these errors consisted of having commas in spots where I should have had semicolons, creating the wrong spacing on the screen. I also played with the program a little bit.

Monday, November 18

Before VAX crashed again, I changed the main screen of the program. I added a new copyright date, took out my name, and the other people's names, and positioned lines nicer on the screen.

Some items were enlarged to fit better too. Also, Ball State University had its name printed by the second copyright.

Tuesday, November 19

I did not work today because of personal reasons.

Wednesday, November 20

I did not work today because of personal reasons.

Thursday, November 21

I did not work today because of personal reasons.

Friday, November 22

I could not work today because vax was down.

Monday, November 25

After running the program several times, I noticed some items were unneeded. For example, at the bottom of some REGIS pictures, it had the left and right arrows, as well as press a letter to go back. Since left arrow and the letter meant the same thing, I removed the move a letter. This made it easier for the user and more consistent.

Tuesday, November 26

One error in the program was that a section was not calling the parts it claimed it would call. I had a problem where the items in my case statement were out of order. This was solved today.

Wednesday, November 27 through Friday, November 29

I did not work these days because of the school recess.

Monday, December 2

Paul ran the program and wrote down errors he found. Then I corrected these errors. One error was that the color commands were printed on the bottom of one of the sixelized screens. I needed to edit the sixelized REGIS file to correct this problem.

Tuesday, December 3

Numbers, lines, and words were placed on two of the United States maps today. These sectioned areas with numbers were considered zones for one of the data sections.

Wednesday, December 4

Today I used the solid world map and put dots on it, as they appeared in the original Hothouse program. These dots showed how much rainfall occurred in each area. Some areas did not appear on the IBM map, so I guessed at how much rain in these areas, such as Greenland.

Thursday, December 5

I ran the program, and put on final touches as needed. The Natural Resources Department will probably have me correct items to better suit their needs. I might also have to add the VT100 section later. Hothouse is finished for the semester though.